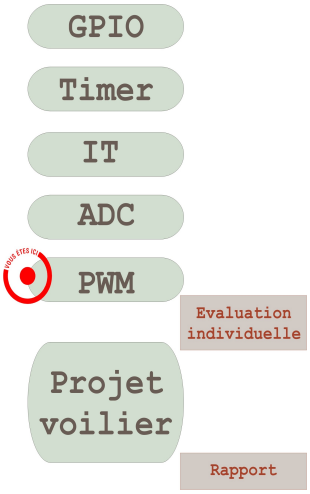
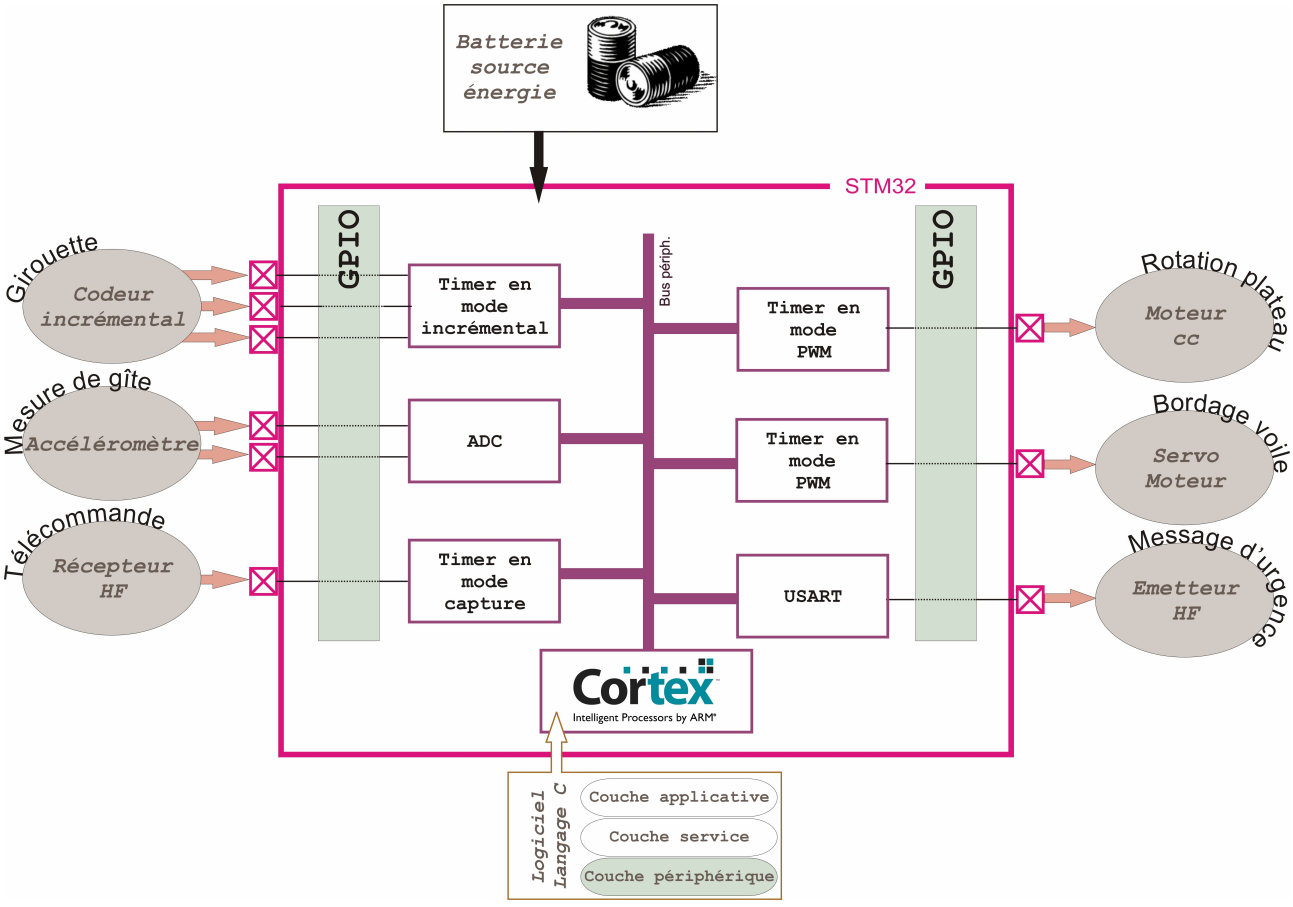


TP *sur la PWM (Pulse Width Modulation)*

1- La PWM dans le système *Bordage de voile automatique d'un voilier*
2- Le PWM en général
3- Travail de l'étudiant – configuration d'un timer en PWM

1 La PWM dans le système *Bordage de voile automatique d'un voilier*

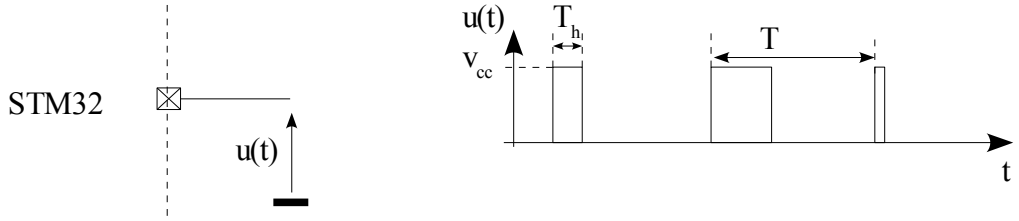


La ***PWM*** (ou ***MLI, M***odulation de ***L***argeur d'***I***mpulsions) intervient dans le projet à deux niveaux très différents :

- commande du servo moteur : dans ce cas, la PWM est utilisée comme un ***format de transmission*** au même titre que l'*AM*, la *FM*... Il s'agit en effet de communiquer une *information* du micro-contrôleur vers le servomoteur, par le biais de la durée d'une impulsion.
- Génération d'une tension analogique : ici, la PWM est associée à un filtre passe-bas de manière à **produire une tension analogique** (cela évite l'utilisation d'un *DAC*).

2 La PWM en général

2.1 Généralités



Le signal u(t) est périodique, et possède donc une fréquence F=1/T. Sa forme est numérique, la durée de l'impulsion notée T_h est susceptible de varier de 0 à T.

2.2 Utilisation en transmission d'information

Dans ce cadre, les deux dispositifs qui vont communiquer (l'émetteur et le récepteur d'information) doivent « s'entendre » sur la fréquence du signal PWM. Une fois cela résolu, le récepteur sera attentif uniquement à la durée de chaque impulsion pour en déduire l'information portée.

L'émetteur (*STM32*) élabore le signal PWM en contrôlant numériquement la largeur de chacune des impulsions. L'étendue de valeurs possibles, finie, constitue la ***résolution*** de la PWM, par exemple :

Fréquence de PWM : 1kHz
Résolution : 256 valeurs possibles pour la durée T_h. Dit autrement, une résolution de 8 bits.

Pour information, on pourra dire de cette transmission qu'elle a les caractéristiques suivantes :

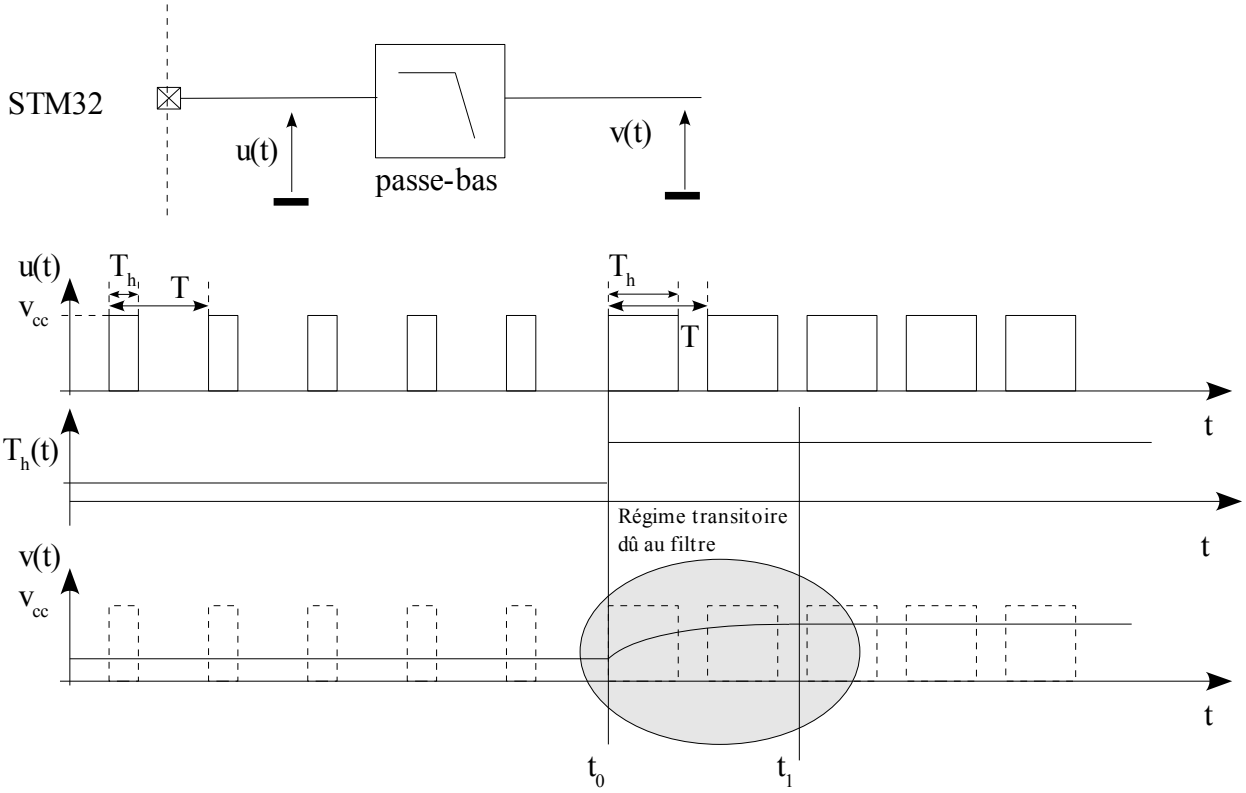
Vitesse de transmission ou débit de moments (débit d'impulsions dans notre cas): 1kBauds
Débit binaire : 8kbits/s ou 1kbyte/s (byte = octet)

Dans le cas qui nous intéresse, le servomoteur travaille à une fréquence de 50Hz ou 40Hz (selon le type). Sa période est donc de 20 ms ou 25 ms. Il est sensible à des largeurs d'impulsions qui sont comprises entre 1 ms et 2 ms environ. Dès qu'il reçoit une impulsion, il traduit sa largeur en un angle mécanique. Autrement dit, pour 1 ms on obtient l'angle minimum de rotation de l'arbre du servomoteur, pour 2 ms l'angle maximum.

2.3 Utilisation pour générer une tension analogique

Dans ce cas, on change totalement la manière de voir la PWM. Bien que véhiculant une information au sens des transmissions, la grandeur qui va nous intéresser au final, n'est plus directement la durée. Nous allons voir en effet, que le signal PWM va être associé à un filtre passe-bas, et ne pourra en être dissociée pour l'obtention de la tension analogique.

Prenons un exemple de signal PWM:



Le comportement entre t_0 et t_1 peut surprendre, mais il traduit bien le que fait le STM32, à t_0, provoque un changement de durée T_h , ce qui s'apparente à un « échelon de durée ».

Analyse :
De t = 0 à t= t_0 (et pour t>t_1) , la durée des impulsions est constante (environ le ¼ de la période, puis ¾). Sur ces intervalles de temps on peut calculer très simplement la valeur moyenne de u(t), elle vaut (de 0 à t_0):

$$u_{moy} = \frac{T_h}{T} . V_{cc} = \alpha . V_{cc} = \frac{V_{cc}}{4}$$, α est le ***rapport cyclique*** de u(t)

Un filtre passe-bas laisse toujours passer la valeur moyenne de la tension d'entrée. Si par ailleurs, sa fréquence de coupure est très basse devant la fréquence PWM, le filtre **va supprimer toutes les ondulations** pour ne **garder que la composante continue**.

Entre t_0 et t_1, on observe un régime transitoire. Plus la fréquence de coupure est basse, plus ce régime est long, ce qui peut être gênant. On a donc un **compromis** à trouver entre **ondulation acceptable**, et **durée du régime transitoire pas trop longue**.

On retiendra :

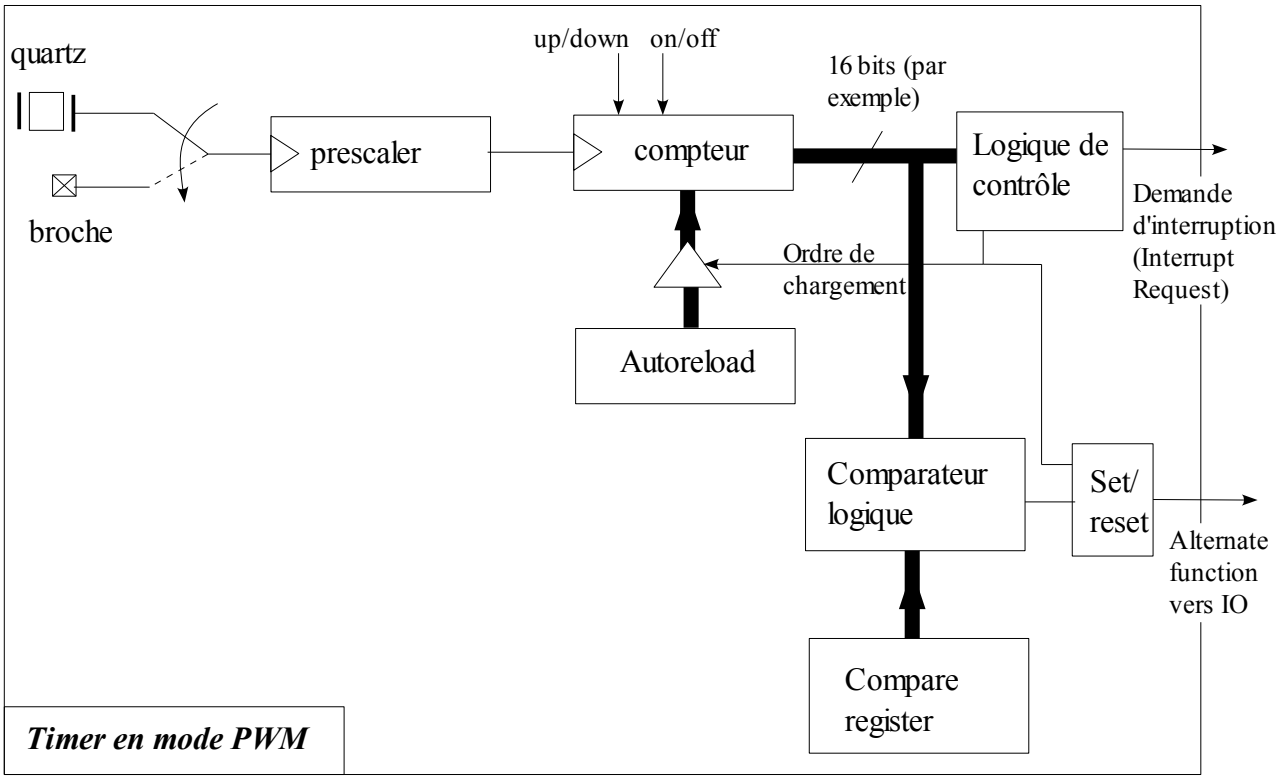
Un signal ***PWM*** est caractérisé par :
- sa ***fréquence***
- sa ***résolution***

Si la PWM est utilisée comme un ***DAC*** (Digital to Analog Converter), c'est à dire pour élaborer une tension analogique, il faut lui associer un ***filtre passe-bas*** de ***fréquence de coupure adaptée*** au cahier des charges, en tout cas faible par rapport à la fréquence PWM.

Habituellement, un rapport de 10 à 20 entre fréquence PWM et fréquence de coupure du filtre passe-bas (RC) est convenable.

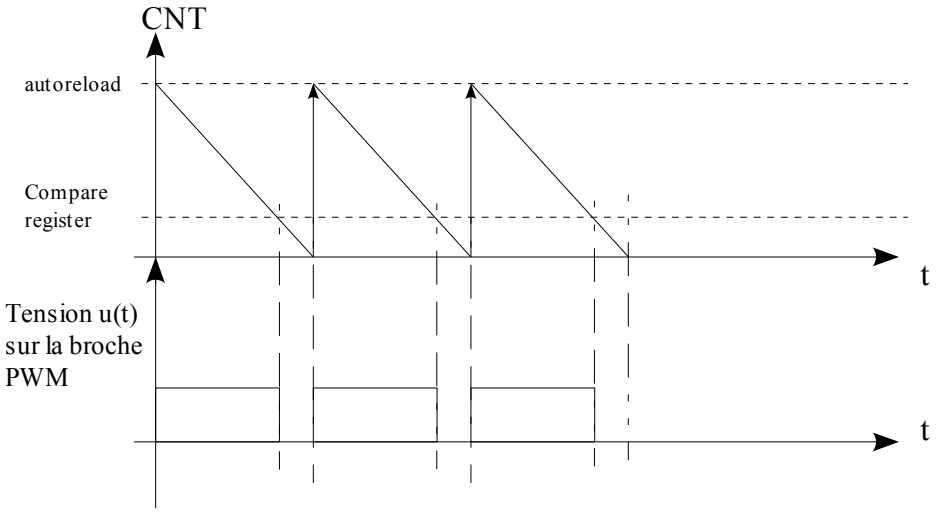
2.4 Structure habituelle d'une unité PWM dans un micro-contrôleur

La PWM utilise systématiquement un Timer auquel on associe un nouveau registre, dit *registre de comparaison (compare register)*, comme le montre le schéma ci-dessous :



Le Timer devra être configuré pour qu'il déborde à la fréquence PWM. Au débordement, une broche dédiée passe à '1' grâce à une logique séquentielle (RS par exemple). Au moment où le Timer atteint la valeur contenue dans le registre de comparaison, alors la broche passe à '0'. On obtient ansi la PWM souhaitée, avec la durée d'impulsion contrôlée par le registre de comparaison.

Chronogramme :



Dans bien des cas, un bit de configuration permet d'inverser la polarité du signal de sortie afin de ranger presque directement dans le registre de comparaison, la durée à l'état haut.

En résumé, 2 paramètres du périphérique Timer en mode PWM vont influer sur les 2 paramètres de la PWM (résolution et fréquence), ce sont :

- la **fréquence d'entrée** du compteur
- la **valeur de l'autoreload**

3 Travail de l'étudiant – configuration d'un Timer en PWM

Pour vous aider :

1. Comment configurer l'entrée/sortie d'un périphérique Timer, Ocy en sortie (y de 1 à 4, puisque 4 entrées/sorties par Timer) ?
2. Quel bit permet de contrôler la polarité de la sortie y ?
3. Quel bit permet de valider la sortie y ?
4. On propose d'utiliser le mode 1 pour la PWM. De quoi s'agit-il ? Quel champ de bit lui est associé ?

NB: Le timer 1 est plus complexe que les autres. Un bit supplémentaire doit être mis à 1 pour valider la sortie y, c'est le bit *MOE*.

3.1 Configuration du PWM

⌦ Ajoutez une fonction de configuration des Timers en *alternate output push-pull*. A vous de définir proprement son prototype.

⌦ Complétez le pilote *Timer_1234.c*, pour y ajouter une fonctionnalité *PWM*.

```
      vu16 PWM_Init(TIM_TypeDef *Timer, char Voie, float Frequence_PWM_Khz);
//
// Cette fonction initialise la voie spécifiée du Timer voulu en PWM.
// La fréquence souhaitée est passée en paramètre.
// La fonction renvoie un entier qui correspond à la résolution de la PWM pour
// pouvoir ensuite régler les rapports cycliques.
//
//  3 Timer "general Purpose", TIM2, TIM3 et TIM4 + TIM1
//  Chacun d'entre eux dispose de 4 voies de sorties numérotées de 1 à 4
//  Mapping des IO:
//  TIM1_CH1 - PA08     TM2_CH1  - PA0          TM3_CH1 - PA6          TIM4_CH1 - PB6
//  TIM1_CH1 - PA09     TM2_CH2 - PA1           TM3_CH2 - PA7          TIM4_CH2 - PB7
// TIM1_CH1 - PA10      TM2_CH3 - PA2                    TM3_CH3 - PB0          TIM4_CH3 - PB8
// TIM1_CH4 – PA11      TM2_CH4 - PA3           TM3_CH4 - PB1          TIM4_CH4 - PB9

// Exemple de programmation "statique" à compléter...
#define Set_Value_PWM_TIM_2_Ch_1 TIM2->CCR1
#define Set_Value_PWM_TIM_2_Ch_2 TIM2->CCR2
```

Les deux dernières lignes permettent à l'utilisateur du module, de fixer un rapport cyclique, sans utiliser de fonction (gain de temps) et sans faire appel explicitement aux registres des Timers.
Par exemple la ligne :
#define Set_Value_PWM_TIM_2_Ch_1 TIM2->CCR1
associe à l'expression (parlante), *Set_Value_PWM_TIM_2_Ch_1 ,* l'expression pour initié du STM32, *TIM2->CCR1.*

Du coup, dans une couche service, au lieu d'écrire une ligne du genre :
TIM2->CCR1 = 123;

On écrira
Set_Value_PWM_TIM_2_Ch_1= 123;

3.2 Test de la configuration PWM

⌦ Faites varier régulièrement l'intensité lumineuse de la led PB9 (comme la douce respiration d'un MacBook au repos). **Attention**, avant de partir bille en tête, faites proprement sur papier la conception de cette fonctionnalité en représentant les interactions entre les différents acteurs (la PWM seule ne vous permettra pas de réaliser la fonction) et les informations échangées.